VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

WEBOVÝ SYSTÉM PRO ÚČETNÍ FIRMU

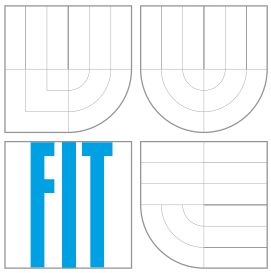
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

RADIM SVOBODA

BRNO 2009

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

WEBOVÝ SYSTÉM PRO ÚČETNÍ FIRMU

WEB SYSTEM FOR ACCOUNTING COMPANY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — RADIM SVOBODA
AUTHOR

VEDOUCÍ PRÁCE — Ing. ŠÁRKA KVĚTOŇOVÁ, Ph.D.
SUPERVISOR

BRNO 2009

Abstrakt

Cílem této práce je vytvořit funkční prototyp informačního systému pro účetní firmu. Systém bude umět zaznamenávat provedené činnosti, pomocí nich vystavovat faktury a zjišťovat pomocí statistických údajů výkonnost zaměstnanců.

Abstract

Scope of this thesis is to create a working protype of inforamtion system for accounting company. The system will be able to record employee's activity, use these records to issue an invoice, and state employee's efficiency using statistics.

Klíčová slova

Informační systém, PHP, MySQL, XHTML, CSS, JavaScript, AJAX, jQuery

Keywords

Information system, PHP, MySQL, XHTML, CSS, JavaScript, AJAX, jQuery

Citace

Radim Svoboda: Webový systém pro účetní firmu, bakalářská práce, Brno, FIT VUT v Brně, 2009

Webový systém pro účetní firmu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením paní Ing. Šárky Květoňové, Ph.D.

.....
Radim Svoboda
19. května 2009

Obsah

Úvod

Neustálý vývoj ve všech odvětvích a oborech si vyžaduje používání moderních metod práce a ty jsou vždy spojeny s požadavky na rychlost, komplexnost a kvalitu.

Informační systém navržený pro účetní firmu sbírá data a informace o činnostech jednotlivých zaměstnanců. Sběr dat napomáhá k objektivnímu posouzení složitosti a náročnosti prováděných úkonů a umožňuje současně i srovnávat výkony jednotlivých pracovníků a lépe podporovat jejich profesní rozvoj.

Řízení firmy v oboru vedení účetnictví vyžaduje neustálý odborný dohled, ale i dohled nad tím, zda práce probíhá v souladu s plánovaným harmonogramem. Je zcela zřetelné, že obor účetnictví prochází prudkým vývojem. Původní plechové bedny s kartami jednotlivých účtů, jež byly typické pro každou účetní, byly vyměněny za počítače, počítače byly postupně řazeny do sítí a v rámci síťového provozu byly zavedeny nové prvky elektronické komunikace mezi firmou, institucemi i klienty. Stále častěji se objevují požadavky účetních na práci z domu. I pro tento případ se dá systém využít. Zavedení informačního systému je nezbytným krokem ve vývoji tohoto oboru a umožní jeho další posun dopředu.

V 1. kapitole je blíže popsán úvod do problematiky. Je zde nastíněna současná situace ve firmě a problémy, které činnost firmy zatěžují. Zároveň tu jsou vyjmenovány požadavky na systém, který by měl firmě usnadnit práci.

V 2. kapitole se analyzují kladené požadavky. Vyhodnocuje se, které entity v systému existují a jaké jsou vztahy mezi nimi. Také jsou zde navrženy některé rysy uživatelského rozhraní.

V 3. kapitole jsou vyjmenovány a stručně popsány jazyky, technologie a nástroje použité při tvorbě této práce, které také budou použity při provozu tohoto informačního systému.

V 4. kapitole je popsána samotná implementace. Popis není kompletní a detailní, protože v takovém případě by byl rozsah této práce příliš velký a pro čtenáře by se práce mohla stát nepřehlednou. Pro bližší pochopení implementace je vhodné si projít zdrojové kódy.

V posledních kapitolách jsou zmíněny možnosti na rozšíření systému, které by mohly uživatelům více usnadnit práci. Na závěr jsou zhodnoceny dosažené výsledky a zmíněny zkušenosti, které jsem při tvorbě této práce získal.

Kapitola 1

Specifikace požadavků

1.1 Současná situace

Firma musí usměrňovat, kontrolovat a řídit prováděné práce. K tomu účelu vytvořila nejprve formulář nazvaný „Výkaz prací“, který se ukázal jako nedostatečný, protože umožňoval značnou kreativitu myšlenek jednotlivých zaměstnanců a zpracování Výkazu prací přinášelo značné problémy pro jeho další využití k fakturaci a oceňování provedených prací. Firma tedy přistoupila k vytvoření nového formuláře nazvaného „Zakázkový list“, který byl vytištěn pro jednotlivé klienty vždy každý měsíc. Z používání tištěných zakázkových listů vyplynula jejich nevýhoda, spočívající v tom, to, že vzhledem ke specifikaci prováděných úkolů bylo nutno provádět průběžnou fakturaci a nečekat až na splnění posledních úkonů, které byly brzděny často nedodáním, či pozdním dodáním podkladů pro zpracování. Nutnost průběžné fakturace vyvolala potřebu pracovat průběžně i se zakázkovými listy tak, že byly průběžně předávány k dílčímu fakturování a vraceny zpět příslušným pracovníkům. Cirkulace zakázkových listů způsobila nepřehled o jejich pohybu. Následně se přistoupilo k tomu, že k jednomu klientovi bylo vydáváno v jednom měsíci více zakázkových listů, aby bylo možno fakturovat dílčí ukončené práce vztahující se ke konkrétnímu měsíci, například pro zpracování mezd a následně pro zpracování DPH a nakonec po zpracování celého účetnictví za uplynulý měsíc. Zlepšení stavu řízení prací s sebou však přinášelo potřebu tisku mnoha formulářů a složitou administrativu s jejich vyplňováním a ukládáním. Spotřeba času na administrativní činnosti spojené s řízením formulářů byla neúměrně vysoká a bylo nutno přistoupit k jejich zefektivnění. Z praxe firmy tedy logicky vyplynuly následující požadavky:

Snížit nepřesnost a chybovost uváděných informací

Spočívá v tom, že nositel informace bude co nejvíce využívat systém výběru z předem definované nabídky a přitom bude mít možnost doplňovat vybranou nabídku o zpřesňující informace v omezené míře. Zabrání se chybám vznikajícím z nečitelnosti písma a nahradí se ručně psané texty výběrem z předem nastavených údajů.

Snížit vysokou spotřebu času, kancelářských papírů a tiskových náplní

Manipulace se zakázkovými listy, jejich průběžné ukládání na vyhrazené místo, potřeba průběžného získávání konkrétních informací – to vše přinášelo značné ztráty způsobené hledáním, urgováním, vypisováním a neustálým posunem dokumentů na místa podle vnitřních

pravidel pro práci s dokumenty. Samotný tisk zakázkových listů rovněž značně zatěžoval režii firmy a následná manipulace s nimi vyžadovala další spotřebu času, který by jinak mohl být využit na výkon odborných činností.

Možnost ztráty ZL

Ne vždy se však podařilo všem pracovníkům důsledně dodržovat předem stanovená pravidla a správně pochopit oběh dokumentů. Někdy bylo nutno tisknout náhradní Zakázkový list, když se původně vytištěný dokument ztratil z oběhu. Následné nalezení „ztraceného“ ZL pak způsobovalo nepřehled v celkovém měsíčním vyhodnocování provedených prací a v archivaci zakázkových listů.

Špatná čitelnost

Škrtání a přepisování chybných údajů způsobovalo špatnou čitelnost a následné chyby. Dokument, který se pohyboval po pracovištích po dobu jednoho měsíce ztrácel svou čitelnost i tím, že na něj působily vlivy způsobené jeho častým používáním. Možnost provádění zápisů ručně pak způsobovala nečitelnost a chyby v následných úkonech, jako například chyba v názvu firmy na vystavené faktuře. Odstraňování těchto nedostatků častými dotazy u zpracovatele zakázkového listu bylo dalším zatěžováním a způsobovalo i vnitřní napětí.

Neefektivní práce s výsledky

Vybráním zakázkových listů od všech pracovníků byl zahájen proces jejich zpracování. Bylo nutno vyspecifikovat práce, které jsou podkladem pro fakturaci a následně vyspecifikovat i všechny ostatní nezbytné práce pro zpracování podkladů pro odměňování zaměstnanců, kteří se zúčastnili na produktu. Práce se zakázkovými listy vyžadovala ruční sčítání údajů a vyhodnocování sdělených informací. Při následných kontrolách byly zjišťovány nepřesnosti zaviněné chybným natypováním nebo nesprávným načtením údajů.

Výše uvedený postup řízení prací vyvolal potřebu provedení zásadních změn jak při evidování prováděných prací, tak při jejich vyhodnocování, spočívající ve vytvoření centrálního informačního systému. Úkolem nového informačního systému bude odstranit co nejvíce shora uvedených problémů a umožnit zpřehlednění průběžných informací pro všechny zainteresované osoby. Požadavek na vytvoření Zakázkových listů v elektronické podobě s možností jejich vyplňování pomocí výběru z předem definované nabídky a případným upřesněním údajů pomocí poznámky se jevil jako jedině možná cesta zefektivnění prací a současně i dosažení sběru a zpracování objektivních informací. Důležitým úkolem je sbírat informace pro vedoucího pracovníka, kterého představují v tomto případě dvě osoby, z nichž má jedna, na základě vnitřně určených kompetencí, za úkol fakturovat provedené práce, a druhá zpracovávat ze všech informací podklady pro odměňování v souladu s vnitřními pravidly pro odměňování.

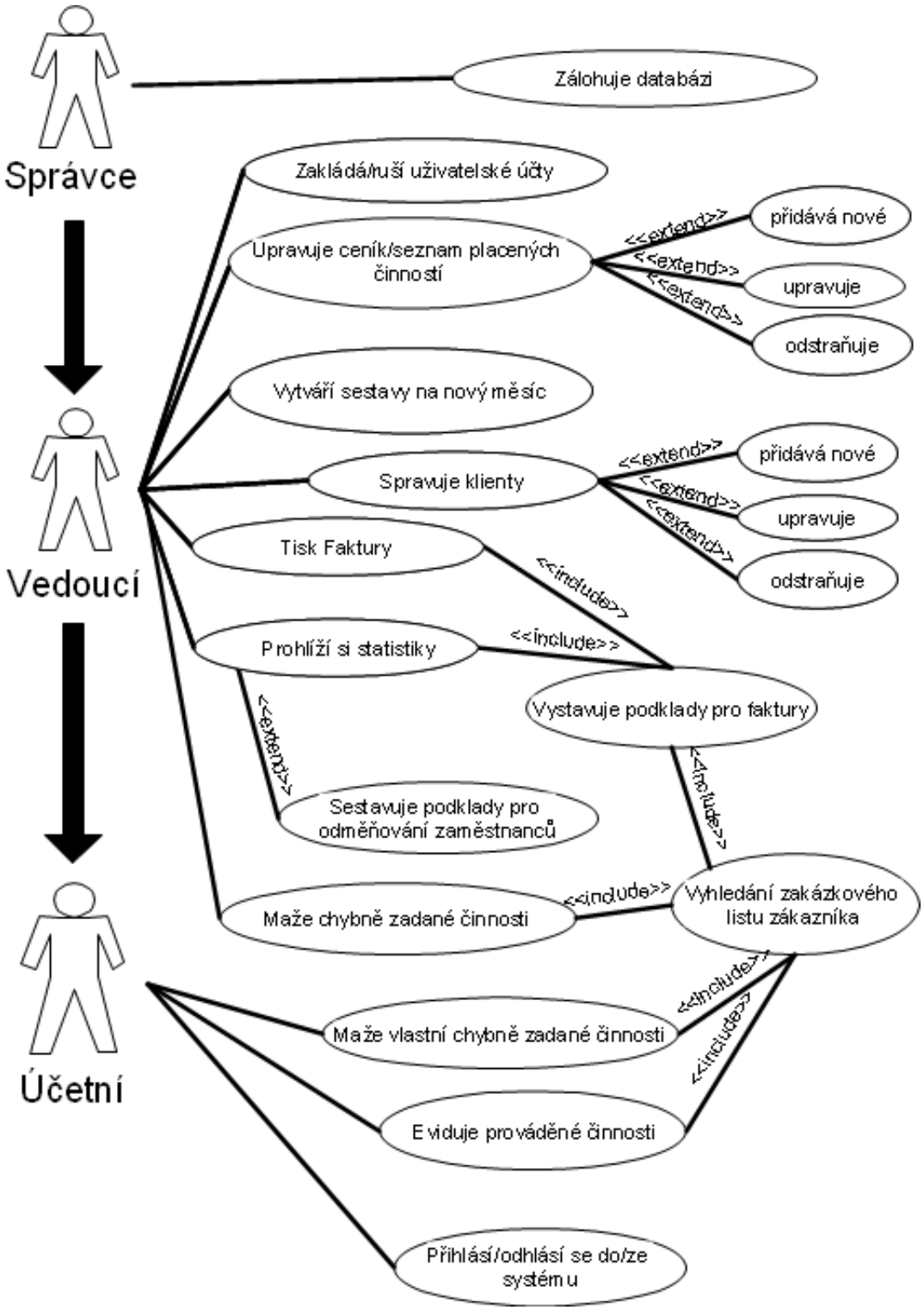
1.2 Uživatelé systému

Systém pro účetní firmu budou používat jen zaměstnanci firmy. Zaměstnance lze rozdělit do tří kategorií a stručně popsat jejich možnosti asi takto:

- **Účetní** – uživateli je přidělen uživatelský účet a ten pak eviduje vykonávané činnosti u různých klientů.

- **Vedoucí** – má možnost využít evidovaná data zaměstanců např. pro vystavovaní faktur nebo prohlížení statistik o efektivitě práce.
- **Administrátor** – účet administrátora umožňuje navíc zálohovat databázi.

Podrobněji jsou požadavky znázorněny na obrázku 1.1 a dále také blíže vysvětleny.



Obrázek 1.1: Diagram případů užití

1.3 Popis diagramu případů užití na obr. 1.1

Vyhledání zakázkového listu klienta

Dříve než začne zaměstnanec zadávat údaje, musí si vybrat příslušný zakázkový list. Ty budou rozděleny do skupin podle měsíců a v každé takové skupině bude seznam zpracovávaných klientů.

Evidence prováděných činností

Po nalezení ZL zaměstnanec vybere konkrétní činnost ze seznamu. Zadá skutečnou spotřebu času na provedení předmětné činnosti, popř. počet např. zpracovaných dokladů. Pokud zaměstnanec konkrétní činnost v seznamu nenajde, nebo je její název příliš obecný, lze ji blíže specifikovat pomocí poznámky. Systém by měl umožňovat přidat ke každé zaznamenané činnosti aktualní čas, tj. čas, kdy k zápisu došlo. Provedený zápis pak pracovník nemůže následně měnit.

Mazání chybně zapsaných činností

Vychází se z toho, že při každé činnosti může dojít k chybě, a tedy i záznamy uvedené do systému mohou obsahovat chybu. Každá účetní má proto možnost svou chybu smazat a uvést informaci správně. Do záznamů provedených jinou účetní však zasahovat nelze. Tuto možnost má pouze vedoucí a provádí mazání až po vysvětlení důvodu chybného záznamu. Smazané záznamy je možno pro lepší přehled následně zobrazit.

Správa klientů

Vedoucí má na starosti správu klientů. Může přidávat nové, odstraňovat nebo upravovat. Důležité také bude rozpoznat, zda je klient pravidelně zpracovávaný nebo jen příležitostně.

Vytváření sestavy na nový měsíc

Každý měsíc je nutné vytvořit novou sestavu, ve které budou klienti, kteří se v daný měsíc zpracovávají. Taková sestava může být každý měsíc jiná. Je třeba odlišit klienty, kteří se zpracovávají pravidelně každý měsíc a pak klienty, kteří se zpracovávají jen příležitostně. Pravidelně zpracovávaní klienti se do takové sestavy přidají automaticky, příležitostní manuálně, tzn. vedoucí je musí přidat ručně.

Správa ceníku/činností

Základem funkčnosti systému je kvalitně zpracovaný ceník, který nejenže popisuje pod jednotlivými kódy všechny možné úkony, ale dává i prostor pro nestandardní postupy a činnosti, které se mohou vyskytnout. Je současně třeba, aby i zaměstnanci byly důsledně obeznámeni s tím, co je náplní jednotlivých ceníkových položek a jak je mají používat. Je přitom zřetelné, že ceník musí mít možnost přijímat změny nejen ve výši stanovené ceny, ale i v obsahu. Musí mít možnost zavádět nové položky a rušit položky, které jsou evidentně nevyužitelné. Správu ceníku má v plné kompetenci vedoucí.

Správa uživatelských účtů

Jedná se o firemní systém, proto se uživatelé nemohou volně registrovat. Vedoucí uživatelské účty zakládá, popř. ruší a přiděluje je zaměstnancům. Zaměstnanc si své heslo může kdykoliv změnit. Není možno, aby údaje do sytému zadával zaměstnanec pod cizím jménem.

Vedení firmy potřebuje průběžně, a tedy i historicky udržovat přehled o tom, který zaměstnanec se v jakém čase a rozsahu podílel na činnosti firmy. S odchodem zaměstnanců však údaje o jejich účasti na procesu zůstávají zachovány.

Faktury

Jedním z nosných výstupů ze systému je vytváření podkladů pro fakturaci. Záznamy ze zakázkových listů má k dispozici vedoucí, který má možnost transformovat údaje zadané ze strany účetních do závěrečného souboru informací, na základě kterých je následně vystavena faktura. Systém umožňuje převzaté informace upravovat, protože tyto informace mohou obsahovat např. gramatické chyby v doplněných poznámkách. Výsledný produkt je pak možno tisknout v tabulce, která je ve formě přiložené rekapitulace součástí faktury. Systém pak identifikuje informace, které již proběhly fakturačním řízením a úkony, které doposud fakturovány nebyly. Tím je zajištěno, že všechny fakturovatelné údaje, které zaměstnanec pořídí, projdou fakturací, nebo zůstávají do doby konečného vyfakturování v evidenci nevyfakturovaných údajů.

Statistiky – podklady pro mzdy

Statistické údaje jsou používany především pro volbu firemní strategie, posuzování změny rozsahu prováděných prací, sledování vývoje rozsahu účasti pracovníků na produktu a současně i pro možnost rozhodování o cenové politice na základě získaných informací. Statistické sledování je proto otevřenou oblastí, jejíž rozsah se bude měnit a upravovat podle interních potřeb firmy.

Kapitola 2

Analýza a návrh

2.1 Entity Relation Diagram

Entity Relation Diagram (ERD) je abstraktní grafické vyjádření dat, která si systém musí pamatovat. Jedná se o konceptuální schéma, v našem případě pro relační databázi[1]. V návrhu informačního systému se tyto modely používají při analýze požadavků. ERD pro náš informační systém je na obr. 2.1 a dále i blíže popsán.

2.2 Popis ERD

Klient

Tato entitní množina znázorňuje klienty, které firma zpracovává. Jednoduchý atribut „Jmeno“ je pro celé jméno klienta, tj. jméno a příjmení, nebo název firmy. Některé klienty firma zpracovává pravidelně, jiné pouze příležitostně – např. jednou do roka. Pro rozlišení takových klientů použiji atribut „flag“. V MySQL lze s výhodou použít datový typ *enum*. Mohu tak klienta označit jako aktivního, příležitostného nebo třeba vyřazeného.

Období

Tato množina uchovává informace o tom, kdo a ve kterém období se zpracovává. Užitečné taky bude vědět, zda už je klient v konkrétním období zpracován nebo se stále zpracovává anebo se zpracovávání zrušilo. Pro tyto účely opět použiji datový typ *enum*.

Ceník

Jak název napovídá, jedná se o ceník a tedy i seznam nabízených služeb. Účetní z tohoto seznamu může vybrat činnost, kterou právě provedla. Vedoucí může pomocí něj snadno vystavovat faktury.

Činnosti

Množina udává, kdo, kdy, co a kolik udělal. Slouží tedy jako přehled prováděné práce a podklad pro faktury.

Poznámky

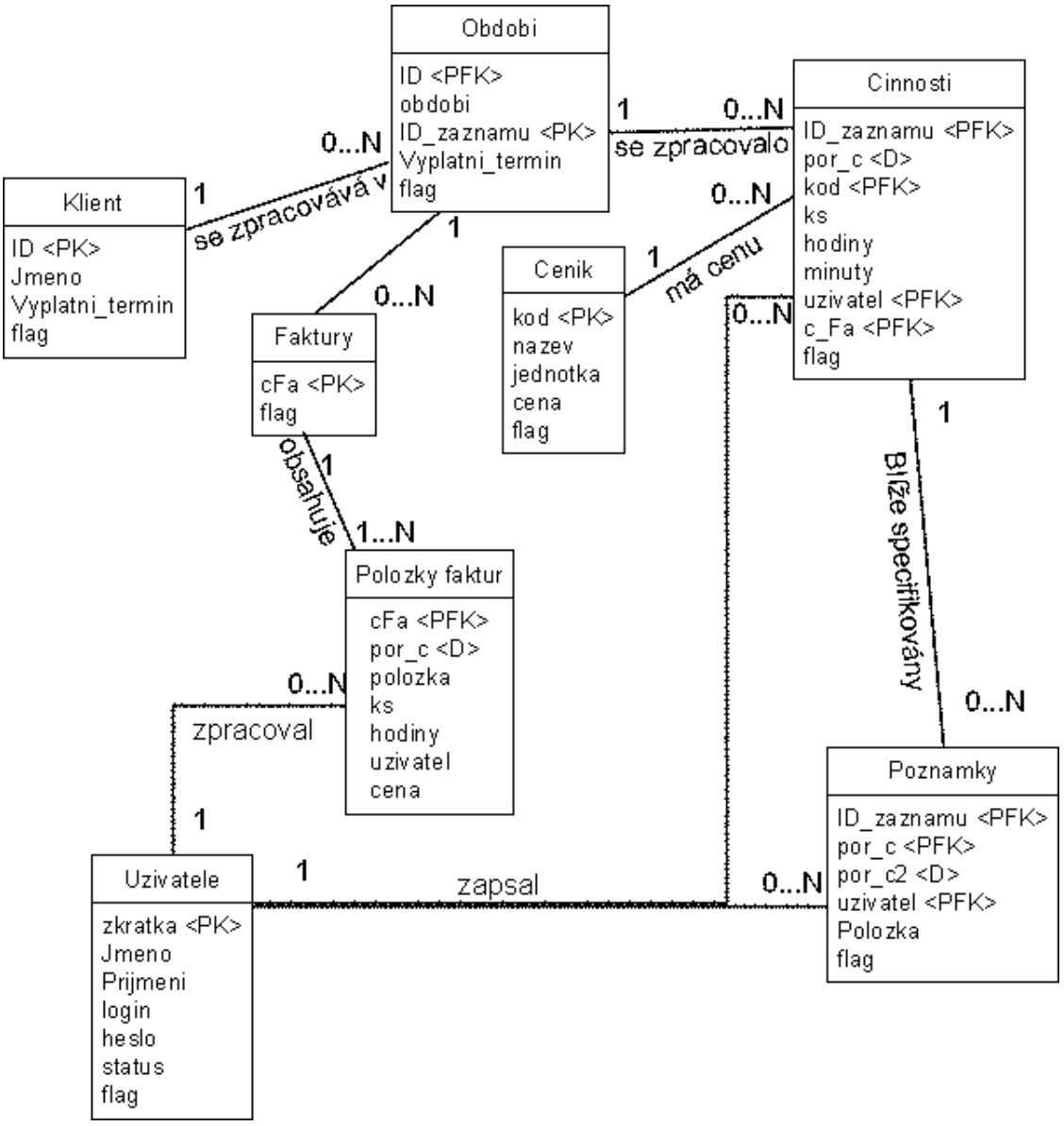
Slouží pro bližší specifikaci provedené činnosti. Může se jednat např. o podání odhlášky z nemocenského pojištění nějakého konkrétního zaměstnance. Pokud by se takových odhlášek podávalo víc, poznámky pomohou udržet přehlednost provedené práce.

Faktury

Uchovává informaci, které faktury a pro koho jsou vystaveny. Užitečné může být si fakturu uložit jako rozpracovanou a později se k ní vrátit, dodělat a vystavit. K tomu využiji atribut „flag“ typu *enum*.

Položky faktur

Jednotlivé položky faktur. Může se zdát, že pro tyto účely již šlo využít entitní množinu *Činnosti*. Protože však s klienty existují smluvní vztahy upravující ceny některých služeb, nedají se *Činnosti*, *Poznámky* a *Faktury* přímo pro fakturaci použít.



Obrázek 2.1: Entity Relation Diagram

2.3 Uživatelské rozhraní

Bude se jednat o webovou aplikaci. Vzhled stránky by měl být jednotný, jednoduchý a ne příliš rušivý.

Navigace

Základem rychlé práce se systémem je jednoduchá a přehledná navigace.

Hlavní menu bude na levé straně. V případě zobrazení spodní části nějakého delšího seznamu, zůstane hlavní menu viditelné.

Další důležitou součástí je **navigace mezi jednotlivými měsíci**. Vhodné bude použít horizontální lištu, kde si uživatel bude moci vybrat, zda jít doleva (předchozí měsíc) nebo doprava (následující měsíc).

Chyby uživatelů

Jak už bylo zmíněno, chybovat je lidské. Aplikace by měla kontrolovat vstupní data odesílána na server, aby se předešlo neočekávanému chování aplikace, nebo dokonce narušení konzistence databáze. V případě útoku samotná kontrola vstupních dat na straně klienta pomocí JavaScriptu nestačí. Je potřeba data zkontrolovat těsně před tím, než dojde k zápisu dat do databáze – tedy na straně serveru.

Rychlost aplikace

Na rychlost aplikace může mít vliv velké množství faktorů jako i kondice serveru nebo síťového připojení. Zaměřím se nyní spíše na faktory, které mohu přímo ovlivnit. Informační systém je webovou aplikací, kde každá stránka je vždy znovu generována na základě požadavku uživatele a ve spolupráci s databází. Nikdy nemůžeme dosáhnout takové rychlosti jako statické stránky, které se nemění.

Můžeme však některé neměnné části kódu, jako deklarace CSS nebo JavaScriptu, uložit do tzv. **externích souborů**. Díky tomu bude kód generované stránky kratší a stránka tak generována rychleji. Další výhodou externích souborů v moderních prohlížečích je, že jakmile se jednou načtou, zůstanou v cachi prohlížeče a znovu už se načítat nemusí.

Dalším problémem může být neustále znovunačítání téže stránky při jen drobné změně – např. úprava jednoho záznamu na stránce s dalšími desítkami záznamů. Tomu se dá předejít pomocí technologie **AJAX**. Ta umožňuje odeslat požadavek na server a zpracovat jeho odpověď, aniž by se musela znovu načítat celá stránka.

Kapitola 3

Použité prostředky

Celý systém je zpřístupněn pomocí webového rozhraní. Webové stránky lze vytvářet nejrůznějšími technologiemi. Kvůli finanční indispozici používám produkty volně dostupné, popsané níže.

3.1 Jazyk UML

UML (*Unified Modeling Language*) je v softwarovém inženýrství grafický jazyk pro vizualizaci, specifikaci, navrhování a dokumentaci programových systémů. UML podporuje objektově orientovaný přístup k analýze, návrhu a popisu programových systémů. UML neobsahuje způsob, jak se má používat, ani neobsahuje metodiku(y), jak analyzovat, specifikovat či navrhovat programové systémy[6].

3.2 Jazyk PHP

PHP (*Hypertext Preprocessor*, dříve *Personal Home Page*) – skriptovací jazyk na straně serveru, který s využitím databáze dynamicky sestavuje webové stránky a následně je posílá klientovi. Počátky PHP spadají až do roku 1994, kdy si Rasmus Lerdorf ve volném čase vybudoval jednoduchý systém pro evidování přístupu na jeho webové stránky. Ačkoliv byl celý systém původně určen pro osobní Rasmusovo použití, zalíbil se i ostatním uživatelům a začali ho používat[11].

Nyní PHP poskytuje programátorovi nesčetné množství vestavěných funkcí pro práci s řetězci, poli, databází aj.[8]. Hlavním důvodem masového rozšíření bude zřejmě volná dostupnost, podpora více platforem nebo velké množství dostupných informačních zdrojů o tomto jazyku.

3.3 Databáze MySQL

MySQL je multiplatformní databázový systém vytvořený švédskou firmou MySQL AB, který existuje i ve volně dostupné verzi. Komunikace s databází, jak už název napovídá, probíhá ve standardizovaném dotazovacím jazyce SQL (*Structured Query Language*) s drobnými modifikacemi[4].

Úložiště dat (storage engine)

MySQL nabízí několik typů úložišť dat[5]. Pro naše využití jsou nejzajímavější tyto dvě:

1. **MyISAM** – předností tohoto úložistě je rychlost zpracování dotazů. Bohužel však nepodporuje transakce pomocí cizích klíčů.

2. **innoDB** – jako jediný typ úložiště mySQL podporuje transakce na základě cizích klíčů. Neumí však pružně pracovat s atributem `auto_increment` u složených primárních klíčů.

Na základě těchto fakt a také dostupnosti úložišť různých webhostingů volím engine **MyISAM**.

Rozhraní

PHP má mnoho funkcí pro komunikaci s MySQL. Někdy je však nutné nebo přinejmenším vhodné nahlédnout na aktuální stav databáze. Pro tyto účely jsem použil webové rozhraní phpMyAdmin. Pomocí phpMyAdmin lze i snadno zálohovat nebo obnovovat databázi [13] .

3.4 Jazyk XHTML

XHTML (*Extensible HyperText Markup Language*) je značkovací jazyk pro tvorbu hypertextových dokumentů v prostředí WWW[7]. Jazyk XHTML byl vytvořen tak, aby se syntakticky blížil jazyku HTML, ale současně odpovídal více striktní syntaxi XML[2]. Pro samotnou stavbu webového dokumentu použiji XHTML 1.0 Strict. Nedisponuje takovým množstvím značek, např. pro změnu řezu písma, jako HTML, ale to v dnešní době řeší CSS.

3.5 Jazyk CSS

Cascading Style Sheets, česky *kaskádové styly*, je jazyk navrhnutý konsorciem W3C. Kaskádové styly umožnují změnu vzhledu či jiných vlastností elementů v dokumentu. HTML pro některé změny využívá speciální značky. CSS je však propracovanější a poskytuje mnohem více možností. Narozdíl od HTML, kde měním vlastnosti elementů pro každý element zvlášť, v CSS mohu definovat určité vlastnosti pro více elementů vybrané pomocí *CSS selectoru*. Hlavním významem kaskádových stylů je oddělení vzhledu dokumentu od jeho stuktury a obsahu.

Nemalou výhodou kaskádových stylů v externím souboru je, že po načtení souboru do cache se soubor při znovunačtení stránky znovu načítat nemusí.

3.6 Skriptovací jazyk na straně klienta

3.6.1 JavaScript

Skriptovací jazyk na straně klienta nabízející interaktivní a snadnější práci s webovou aplikací umožňuje dynamicky upravovat obsah dokumentu. Standardní rozhraní pro přístup k dokumentu v současnosti představuje Document Object Model (DOM) spravovaný konsorciem W3C. Z JavaScriptu vychází technoligie AJAX.

3.6.2 AJAX – *Asynchronous JavaScript and XML*

Poprvé se objevil v roce 2005. Předností této technologie je možná komunikace webové aplikace se serverem bez nutnosti znovunačítání stránky. JavaScript pošle požadavek na server pomocí `XMLHttpRequest object`, server odpoví a JavaScript zpracuje obdržená data. Celý proces probíha na pozadí. Data od serveru bývají ve formátu XML, ale nemusí to být pravidlem. Zajímavostí je, že AJAX může posílat více požadavků naráz, resp. může poslat další požadavek dříve, než dostane odpověď na první.

3.6.3 jQuery

Pro implementaci výše zmíněných technologií jsem použil volně dostupný JavaScriptový framework jQuery[12]. Tato nadstavba umožňuje snadnější manipulaci s prvky dokumentu a jeden kód je, na rozdíl od standardního JavaScriptu, kompatibilní téměř se všemi webovými prohlížeči dnešní doby.

Pro vyhledání některých elemetnů se v samotném JavaScriptu někdy musí složitě kombinovat metody `getElementById()`, `getElementsByClass()` apod.. JQuery umí vyhledat jeden nebo více prvků snadno pomocí CSS selectoru:

```
jQuery('#main > p.comment')
```

Kapitola 4

Implementace

4.1 Konfigurační soubor

Je vhodné mít pro aplikaci konfigurační soubor. To umožní některé parametry změnit na jednom místě a není nutné procházet celý zdrojový kód, aby se člověk dovtípil, co a kde má vlastně změnit.

Pro tyto účely jsem zvolil přímo soubor *index.php*. V případě, že v databázi rozšíříme tabulku o jeden či více sloupců a informaci, kterou obsahují, chceme zobrazit uživateli, snadno tak provedeme v tomto konfiguračním souboru. Také zde lze nastavit pravidlo pro kontrolu vstupů uživatele ve formě regulárního výrazu.

4.2 Cizí kód

Informační systém je rozsáhlá aplikace. Při jeho realizaci se stále objevují nové a neočekávané problémy, a to může v důsledku znamenat, že na některé úkony máme mnohem méně času, než jsme zamýšleli. V rámci úspory času a snad i kvalitnější implementace je vhodné použít cizí kód, který nám pomůže práci usnadnit.

jQuery

JQuery je JavaScriptový framework řídící se heslem „write less, do more“. Je to jednoduchý a mocný nástroj. Pochopit způsob zápisu není nic závratně složitého. Tento framework nám umožní snazší přístup k prvkům webové stránky, manipulaci s nimi, vytváření nových apod.. Samozřejmě nechybí ani podpora technologie AJAX, práce s událostmi jednotlivých elementů nebo dokonce i animace.

JQuery je dostupná pod licencí MIT nebo GPL. Licence MIT umožňuje jQuery kód kopírovat, upravovat, dokonce i prodávat pod podmínkou, že ve zdrojovém kódu bude vždy zmíněno, že se jedná o jQuery.

jQuery User Interface

Z jQuery vychází jQuery UI[14]. JQuery UI obsahuje moduly pro zajímavé a interaktivní uživatelské rozhraní. Použil jsem moduly dva:

1. **Tabs** – umožňuje rozdělit jednotlivé sekce webové stránky do záložek a učinit ji tak přehlednější.

2. **Draggable** – umožní konkrétním elementům být přesouvány pomocí myši, tak jako je to bežné provádět s okny v dnešních operačních systémech v grafickém prostředí.

Google Chart API

U statistických údajů je pro uživatele přijemné mít tyto údaje shrnuty v přehledném grafu. Pro tyto účely jsem použil volně dostupnou službu Google Chart API[9]. Stačí zaslat požadavek na příslušný webový server, ten požadavek zpracuje a vrátí graf ve formě obrázku.

4.3 PHP skripty

PHP skripty zpracovávají požadavky klientů, komunikují s databází, generují dokument, a ten je zaslán klientovi. Požadavky klientů jsou zasílány metodou GET nebo POST[10]. Popíši několik PHP tříd, které jsem implementoval.

Třída `Page`

Všechny zobrazované stránky v systému mají podobný vzhled – obsahují hlavní menu, odkaz pro odhlášení apod.. Z tohoto důvodu jsem implementoval třídu `Page`. Pomocí jejích metod nastavím, které CSS a JavaScript soubory mají být do dokumentu importovány. Samozřejmě nedílnou součástí je nastavit *callback*, tj. odkaz na nějakou metodu nebo funkci, která se v průběhu zpracovávání kódu vykoná. Tento *callback* vytiskne samotný obsah stránky, tedy informace, které uživatel vyžaduje.

Třída `SimpleTable`

Pracuje s tabulkami databáze, které mohou existovat samostatně a nejsou závislé na žádných jiných. Stačí nakonfigurovat, s kterou tabulkou z databáze a s kterými jejími sloupci se bude pracovat. Tato třída umožňuje s tabulkou provádět základní operace jako zobrazit tabulku, přidat nové záznamy, upravovat je nebo je mazat.

4.4 JavaScript

Pro zajímavější a rychlejší práci s aplikací jsem použil JavaScript, resp. jeho nádstavbu jQuery. Každá zobrazovaná stránka v systému pracuje jinak, proto má většina z nich svoje JavaScriptové kódy. JQuery má implemetováno spoustu metod pro práci s prvky dokumentu. Může se však stát, že potřebujeme metodu vlastní, která bude dělat přesně to, co chceme. JQuery má možnost poměrně snadno přidat další metody. Implemetoval jsem několik metod pro usnadnění práce s formuláři.

Metody pro práci s formuláři

`disableForm(exp,bool)` a `enableForm(exp,bool)` – metody pracují s jQuery objektem, který obsahuje množinu prvků dokumentu. Metoda zkontroluje jednotlivé prvky. Pokud se nejedná o formulářový prvek jako `input`, `select` apod., je ignorován. Formulářové prvky se pak filtrují podle výrazu `exp`. Na základě druhého parametru `bool` se z množiny odstraní ty prvky, které výrazu nevyhovují, resp. vyhovují. S výslednou množinou prvků se pracuje tak, že pro každý prvek je nastaven, resp. odstraněn atribut `disabled`.

Tyto metody jsem využil při odesílání formulářů pomocí AJAX. V praxi se stává, že netrpělivý uživatel stiskne tlačítko pro odeslání podruhé a ve výsledku se v databázi objeví zadávaný údaj dvakrát. S těmito metodami se to stát nemůže, protože formulář je během čekaní na odpověď ze serveru zablokován.

4.5 Registrace uživatelů

Jedná se o informační systém pro zaměstnance relativně malé firmy. Proto není žádná registrace volně přístupná. Nové uživatele registruje vedoucí. Uživatelé si své heslo mohou kdykoliv změnit.

V případě, že uživatel své **heslo zapomene**, vedoucí může uživateli heslo zrušit a zadat heslo nové.

4.6 Přihlášení uživatelů

Uživatelé před vstupem do systému prokáží svoji identitu pomocí jména a hesla. Kvůli bezpečnosti není samotné heslo nikde v databázi uloženo. Při uložení hesla uživatele se používá jednocestná hashovací funkce – funkce `PASSWORD()` v MySQL. Pokud by se útočníkovi podařilo získat data o uživatalích z databáze, bude pro něj velmi obtížné získat skutečná hesla, zvláště pokud bude heslo silné a nebude se nacházet v běžném slovníku.

```
mysql > SELECT PASSWORD( 'abc5' );
        *0CE2D75AFD05A23559F79AE8A3FF7F04AC7E5C65
```

Relace uživatelů se serverem je udržována pomocí superglobálního pole `$_SESSION` na straně serveru. Aby se neustále nemuselo komunikovat s databází, pole `$_SESSION` uchovává informace o tom, kdo je přihlášen (ID uživatele) a jaká má přístupová práva.

4.7 Systémové požadavky

Aplikace vyžaduje na serveru, na kterém běží, tyto technologie:

- PHP 5 nebo vyšší s podporou SESSION
- MySQL 5 nebo vyšší
- Apache 1.3 nebo vyšší

Na straně klienta je potřeba mít moderní prohlížeč s podporou JavaScriptu a cookies. Testováni probíhalo v prohlížečích Firefox 2, Firefox 3, Internet Explorer 7.

4.8 Testování

Systém je vyvíjen pro webový prohlížeč Mozilla Firefox a dále testován na Internet Explorer 7 (IE). Testování bylo zaměřeno na funkčnost a bezpečnost aplikace. Testovaní jsem prováděl já a také zadavatel této práce, aby se dosáhlo objektivnějších výsledků.

Funkčnost

Aplikace byla v obou prohlížečích funkční. Problém byl pouze s vypisováním kontrolních hodnot pomocí `console.log()`, který Internet Explorer neznal. Všechny kontrolní výpisy jsem tedy smazal nebo přinejmenším zakomentoval. Aplikace nyní běží, jak má. V IE je pár odlišností ve vzhledu dokumentu, taková závada by se však měla dát odstranit ne příliš složitým způsobem.

Bezpečnost

Bezpečnost je u internetových aplikací důležitou součástí. Tento systém má však být pouze pro interní potřeby firmy, napadení útočníkem tak není příliš pravděpodobné. Během testování se nepodařilo provést nedovolené úkony – úkony, ke kterým nemá uživatel dostatečná oprávnění. Ani se nepodařilo pozměnit data v databázi jiným než dovoleným způsobem.

Kapitola 5

Možnosti rozšíření

Prototyp informačního systému splňuje požadavky, které na něj byly zadavatelem kladeny. Z používání systému v praxi vyplyne, jaké další možnosti by mohl systém nabízet, popřípadě jak by se měl systém změnit pro účely jednoduššího používání uživatelem. Požadavky na možnost rozšíření vyplývají především z toho, že činnosti v oblasti vedení účetnictví se vyvíjejí v souladu s vývojem legislativy i techniky a současně i na základě potřeby neustálého zefektivňování prováděných činností.

Už teď ale vím, že by šlo implementovat hned několik rozšíření, která by uživateli usnadnila práci s aplikací.

5.1 PDF

Uživatel může různé sestavy, které mu systém nabízí, tisknout. Pokud chce uživatel nějakou sestavu uložit na disk ve formátu PDF, musí pro to použít externí aplikaci. Na internetu existuje řada knihoven pro PHP, které umožňují generovat dokumenty PDF. Některé produkty jsou komerční, jiné ne. Problémem je však fakt, že většina takových nekomerčních knihoven nepodporuje diakritiku. Zvolil bych knihovnu **FPDF**, která je volně dostupná a navíc podporuje kódování ISO-8859-2[3], tzn. podporuje českou diakritiku.

5.2 Doplňující informace

Někteří klienti se mohou lišit od ostatních tím, že mají vyjednané specialní podmínky. Bylo by vhodné systém obohatit o doplňování dalších informací o klientech, jakými jsou např. změny smluvních ujednání, změny podmínek podnikání, speciální požadavky klientů a zpřesňující data, která ovlivňují účetní postupy. Veškeré doplňující informace by měly podle potřeby vstupovat do systému a současně by měla existovat možnost nepotřebné informace ze systému odstraňovat.

5.3 Nápověda

Aplikace nyní neposkytuje žádnou nápovědu pro uživatele. Aplikace je sice vyvíjena tak, aby ovládaní bylo intuitivní, ale může nastat situace, kdy si uživatel s něčím nebude vědět rady. Pro takové případy by existovala nápověda, kde by bylo popsáno co a jak se používá. Zastupovala by tak i uživatelský manuál.

5.4 Zálohovaní a údržba

Zálohovat nyní může administrátor pomocí nástroje phpMyAdmin. Bylo by vhodné naimplementovat možnost zálohování, resp. obnovy dat přímo do systému.

S tím je spjatá i údržba databáze. Stará již nepotřebná data z databáze by se po záloze měla odstranit. Protože však používám datové úložiště *MyISAM*, které nepodporuje práci s cizími klíči, je nutné odstraňovat záznamy „chytře“, aby byla zachována konzistence databáze.

Kapitola 6

Závěr

Prototyp informačního systému byl vytvořen dle představ zadavatele. Systém je vytvořen tak, aby se daly snadno přidávat další moduly dle potřeb, které ukáže až praxe v budoucnosti.

Skriptovací jazyk PHP, který už jsem dříve trochu znal, mi ukázal, že jeho možnosti jsou daleko větší, než jsem tušil. Navíc si nejsem úplně jist, jak velkou část tohoto jazyka jsem stačil prozkoumat.

Framework jQuery, s kterým jsem se poprvé seznámil v předmětu „Tvorba uživatelského rozhraní“ mi ukázal úplně nový pohled na psaní JavaScriptového kódu. Heslo „write less, do more“ (česky „piš méně, udělej více“), které je k vidění na webových stránkách jQuery[12], opravdu platí. Při tvorbě této práce jsem se navíc seznámil s tvorbou rozšíření, resp. vlastních metod pro jQuery, což může být v některých případech velice šikovné a užitečné.

Tvorba tohoto systému pro mě byla přínosem jak ve fázi návrhu a analýzy, tak ve fázi implementační. Jednak jsem získal zkušenosti s jednotlivými programovacími jazyky, ale také jsem se účastnil několika jednání s firmou o jejích pažadavcích na systém, což někdy může být náročnější, než se zdá. Věřím, že firma tento systém využije a ten bude dále vyvíjen dle aktuálních potřeb firmy.

Literatura

[1] Entity-relationship model. [online].
URL http://en.wikipedia.org/wiki/Entity-relationship_model

[2] The Extensible HyperText Markup Language. [online].
URL http://www.w3.org/TR/xhtml1/

[3] FPDF Library. [online].
URL http://www.fpdf.org/

[4] MySQL. [online].
URL http://cs.wikipedia.org/wiki/MySQL

[5] MySQL Documentation. [online].
URL http://dev.mysql.com/doc/

[6] Unified Modeling Language. [online].
URL http://cs.wikipedia.org/wiki/UML

[7] Extensible HyperText Markup Language. 2009, [online].
URL http://cs.wikipedia.org/wiki/XHTML

[8] Mehdi Achour, Friedhelm Betz, aj. : PHP Manual. [online].
URL http://www.php.net/manual/

[9] Google: Google Chart API. [online].
URL http://code.google.com/apis/chart/

[10] Hugh E. Williams, David Lane: *PHP a MySQL - Vytváříme webové databázové aplikace.* Computer Press, 2002, ISBN 8072267604.

[11] Jiří Kosek: *PHP - tvorba interaktivních internetových aplikací.* Grada Publishing, první vydání, 1998, ISBN 80-7169-373-1.

[12] John Resig, jQuery Team: jQuery - JavaScript Library. [online].
URL http://jquery.com/

[13] Marc Delisle: *phpMyAdmin - efektivní správa MySQL.* Zoner Press, 2004, ISBN 8086815099.

[14] Paul Bakaus, jQuery UI Team: jQuery User Interface. [online].
URL http://jqueryui.com/